PROBLEMAS DE PLANIFICACIÓN PARA MÁQUINA ÚNICA EN ENTORNOS DINÁMICOS IMPLEMENTADOS CON METAHEURÍSTICA ACO Y AES

de San Pedro M.
Laboratorio de Tecnologías Emergentes (LabTEm)
Proyecto UNPA-29/B084/1[1]
Unidad Académica Caleta Olivia - Universidad Nacional de La Patagonia Austral
(9011) Caleta Olivia – Santa Cruz - Argentina
e-mail: edesanpedro@uaco.unpa.edu.ar

Leguizamón, G.
Laboratorio de Investigación y Desarrollo en Inteligencia Computacional (LIDIC)
Departamento de Informática
Universidad Nacional de San Luis
(5700) San Luis – Argentina
e-mail: legui@unsl.edu.ar

RESUMEN

Existen dos conceptos de importancia en el contexto de problemas dinámicos y en particular de *scheduling* dinámicos. La búsqueda de soluciones robustas y flexibles. El concepto de robustez de las soluciones se refiere a un tipo de soluciones que pueden ser usadas de igual manera cuando se produce un cambio en el entorno y manteniendo su calidad relativa. En el caso de flexibilidad se refiere a la posibilidad de que las soluciones encontradas puedan ser adaptadas sin mayores problemas cuando se produzca un cambio en el entorno. En consecuencia, soluciones robustas y flexibles son altamente deseables en este tipo de contexto.

Los problemas de planificación dinámicos en entornos de máquina única, han sido encarados principalmente con Algoritmos Evolutivos [3], [4], [5], [6], [7] y [8]. La metaheurística *Ant Colony Optimization* (ACO) a través de numerosos enfoques algorítmicos, fue aplicada con éxito para una variedad de problemas de optimización [11], [12], [13], [14], [15], [16] y [17].

Esta línea de investigación, pretende definir los escenarios dinámicos necesarios, para implementar las metaheurísticas ACO y AEs, en diferentes problemas de *scheduling* para máquina única (*Weighted Tardiness, Average Tardiness, Weighted Number of Tardy Job*), y realizar un análisis comparativo de la calidad de los resultados obtenidos.

1. INTRODUCCIÓN

Dentro de los grupos de investigación enfocados a los problemas de *scheduling*, generalmente se han encarado los modelos conocidos como estáticos, en donde las actividades, los recursos, los tiempos de procesamiento están predefinidos, es decir que no se modifican durante el proceso. Pero en los problemas del mundo real existen otra serie de decisiones que interactúan con el modelo clásico, como por ejemplo cambiar la cantidad y/o configuración interna de los recursos mientras el proceso de *scheduling* está en avance para balancear los cambios en la carga de los *jobs* que arriban al sistema; o puede aumentar o disminuir la cantidad de operarios en el sistema, en distintos momentos del día; o uno o más recursos pueden deshabilitarse temporalmente por razones de falla o

[1] El Grupo de Investigación cuenta con el apoyo de la Universidad Nacional de La Patagonia Austral.

mantenimiento. Si los modelos contemplan alguna de estas decisiones se obtienen los modelos dinámicos [10].

Considerando los distintos escenarios que se pueden presentar en problemas de *scheduling* dinámico, y que tienen que ver con problemas reales en general, se pueden enumerar diferentes aspectos que tienen que ver con:

- Cambio de fecha límite de terminación de las tareas (*due dates*).
- Cambios en el tiempo de procesamiento de una tarea.
- Pueden agregarse nuevas tareas durante el proceso de asignación de recursos.
- Eliminación de tareas que ya han sido planificadas.
- Necesidad de repetir alguna tarea ya procesada.
- Restricción de ejecutar una tarea antes o después de alguna otra determinada.

Estos tipos de cambios pueden ser sencillos en algunos casos, y otros son más complejos de manejar aunque más cercanos a situaciones encontradas en el mundo real. Otros sin embargo, necesitan además tener en cuenta ciertas restricciones que deben contemplarse al momento del proceso de planificación.

En la siguiente sección se describen características de metaheurística ACO y AEs, en la sección 3 se detallan aspectos particulares de ambientes dinámicos y finalmente en la sección 4 se presentan las líneas de investigación y trabajos futuros.

2. METAHEURÍSTICAS ACO Y AES

Todas las metaheurísticas tienen en común que intentan evitar la generación de soluciones de pobre calidad [9], introduciendo mecanismos generales que extienda el problema específico, algoritmos de una corrida como heurísticas de construcción, o búsqueda local de mejora iterativa. Las diferencias entre las metaheurísticas disponibles tienen que ver con las técnicas empleadas para evitar que se estanque en soluciones sub-optimas y el tipo de trayectoria seguida en el espacio de cualquiera de las soluciones, parcial o total.

Una primera distinción que puede hacerse entre metaheurísticas, es si ellas se basan en búsqueda constructiva o local. Otra distinción importante está dad en que en cada iteración, ellas manipulan una solución simple o una población de soluciones. Aunque las metaheurísticas constructivas y basadas en la población pueden ser usadas sin recurrir a la búsqueda local, muchas veces su rendimiento puede ser mejorado enormemente si ésta es incluida. Este es el caso para ACO y AE.

Otra clasificación de metaheurísticas tienen que ver con el uso de memoria, es decir, aquellas que explotan memoria para dirigir la búsqueda futura: *Tabu Search* memoriza explícitamente, soluciones encontradas previamente o componentes de soluciones vistas previamente; *Guided Local Search* (GLS), almacena penalidades asociadas con componentes soluciones para modificar la función de evaluación de soluciones; y *Ant Colony System* (ACO) usa feromona para mantener una memoria de experiencias pasadas.

Es interesante notar que, para todas las metaheurísticas, existe un criterio de terminación no general. En la práctica, se usan un número de reglas generales: el máximo tiempo transcurrido en CPU, el número máximo de soluciones generadas, el porcentaje de desviación desde un *lower/upper bound* al óptimo, y el máximo número de iteraciones sin mejora en la calidad de la solución, son ejemplos de tales reglas. En algunos casos, se pueden definir reglas generales dependientes de la metaheurística.

ACO tiene varias características que en su combinación particular lo hace un enfoque único: usa una población (colonia) de hormigas que construye soluciones explotando una forma de memoria indirecta llamada feromona artificial.

Considerando las características de los ambientes dinámicos, sería deseable contar con un algoritmo de optimización que sea capaz de ir adaptando continuamente la solución a los cambios del entorno, re-usando la información obtenida en el pasado. Los Algoritmos Evolutivos (AEs) parecen ser los candidatos apropiados ya que tienen mucho en común con la evolución natural, y la adaptación en la naturaleza es un proceso continuo.

El problema principal con los AEs, es que ellos convergen eventualmente a un óptimo y pierden así su diversidad que es necesaria para explorar eficientemente el espacio de búsqueda. Así, una vez que la población del algoritmo evolutivo converge, ésta también pierde su habilidad para adaptarse a un cambio en el entorno cuando tal cambio ocurre. En consecuencia, se requiere de mecanismos adicionales que provean permanente diversidad en la población sin perturbar el proceso de búsqueda.

Por otro lado, la metaheurística ACO está inspirada en el comportamiento de las hormigas reales. Puede ser aplicada a cualquier problema de optimización, para que un procedimiento de construcción de una solución pueda ser realizado. ACO se caracteriza por ser un método de búsqueda distribuida, estocástica y basada en la comunicación indirecta de una colonia artificial de hormigas, transmitida por trayectos artificiales de feromona. Estos trayectos sirven como información usada por las hormigas para construir probabilísticamente soluciones al problema bajo consideración. Las hormigas modifican los trayectos de feromona durante la ejecución del algoritmo para reflejar su experiencia de búsqueda [9].

3. CARACTERÍSTICAS DE AMBIENTES DINÁMICOS EN PROBLEMAS DE *SCHEDULING*

Un proceso de *scheduling* implica seleccionar y secuenciar actividades tal que ellas cumplan uno o más objetivos y satisfagan un conjunto de restricciones del dominio del problema. Durante este proceso se deberá seleccionar entre *schedules* (planes o planificaciones) alternativos y asignar recursos y tiempos a cada actividad de manera tal que dichas asignaciones respeten las restricciones temporales de las actividades (*jobs*) y las capacidades limitadas de un conjunto de recursos compartidos, de manera que ciertas funciones objetivo (por ejemplo *tardiness*, *makespan*, etc.) sean minimizadas.

En general, dentro del ámbito de *scheduling*, los modelos más estudiados fueron los modelos conocidos como estáticos u *off-line*, es decir, donde las actividades, los recursos, los tiempos de procesamiento están predefinidos, no se modifican durante el proceso, y con un objetivo involucrando la minimización del tiempo de finalización y los costos de operación. Pero en los problemas del mundo real existen otra serie de decisiones que interactúan con el modelo clásico, como por ejemplo cambiar la cantidad y/o configuración interna de los recursos mientras el proceso de *scheduling* está en avance para balancear los cambios en la carga de los *jobs* que arriban al sistema; o puede aumentar o disminuir la cantidad de operarios en el sistema, en distintos momentos del día; o uno o más recursos pueden deshabilitarse temporalmente por razones de falla o mantenimiento. Si algunas de estas decisiones se adicionan a dicho modelo se obtienen los modelos de *scheduling* dinámicos u *on-line*.

Los modelos de este tipo implican una reconfiguración interna dinámica del proceso de *scheduling* para adaptarlo a la nueva situación del contexto. También pueden existir causas externas que necesiten una reconfiguración, por ejemplo cambio, por parte de los clientes, en las fechas de entrega en función de sus *stocks* y demandas [1]. Uno de los problemas más importantes de *scheduling* es el de máquina única que consiste en el secuenciamiento de un conjunto de tareas para ser procesadas por un único recurso. El estudio de este tipo de problemas es importante porque, a menudo, aparece embebido en un problema de *scheduling* más complejo [2]. Por ejemplo, en un entorno de planificación de tareas sobre múltiples máquinas existe una máquina crítica, cuello de botella, cuya capacidad de procesamiento es más lenta que el requerido por el resto. Un tratamiento de la máquina crítica como un problema de *scheduling* de máquina única puede mejorar notablemente la solución del problema más complejo

4. LÍNEAS DE INVESTIGACIÓN Y TRABAJOS FUTUROS

Actualmente se están realizando diferentes alternativas de la metaheurística ACO, para evaluar el comportamiento de la misma a través de la variación de parámetros, que permita optimizar la calidad de los resultados que brinda el algoritmo [19].

Los trabajos presentados hasta este momento por el grupo de investigación, para problemas de *scheduling* en ambientes dinámicos, se han abordados a través de Algoritmos Evolutivos [5], [6], [7], [8], [18]. Se están analizando distintos escenarios dinámicos con el objetivo de aplicar las metaheurísticas ACO y AE, y que nos permita poder realizar la evaluación del comportamiento de las mismas ante las diferentes alternativas de dinamismo.

Hasta el momento, se están realizando algunos experimentos en ambas metaheurísticas, incorporando a los algoritmos algún proceso de búsqueda local que contribuya a mejorar la calidad del algoritmo que quede reflejado en los resultados obtenidos.

A futuro se prevé el diseño e implementación de ambas metaheurísticas en ambientes dinámicos del mundo real, a los fines de poder realizar una comparación de resultados que nos indique cuál es la mejor heurística a utilizar para el/los modelo/s dinámico/s planteado/s.

Todos estos problemas han sido desarrollados sin restricciones, por lo que en enfoques futuros se incorporarán a estos problemas, algunas restricciones que hasta el momento no han sido analizadas tales como *preemption* vs *nonpreempion*, inserción vs no inserción de tiempo ocioso, *set up* dependientes de la secuencia, dependencia entre jobs (*precedence constrainsts*) entre otros.

AGRADECIMIENTOS

Agradecemos a la Universidad Nacional de la Patagonia Austral por su apoyo al grupo de investigación, la cooperación y las críticas constructivas proporcionadas por el mismo.

BIBLIOGRAFÍA

[1] Morton T.E., Pentico D.W. – *Heuristic Scheduling Systems* – John Wiley & SoonsInc., New York, 1993.

[2] Baker, K. R. – *Introduction to Sequencing and Scheduling* – John Wiley & Sons Inc., New York, 1974.

[3] Madureira A., Ramos C., do Carmo Silva, Silvio – *A Genetic Approach to Dynamic Scheduling for Total Weighted Tardiness Problem*, PLANSIG'99, 18th Workshop of the UK Planning and Scheduling Special Interest Group, Manchester, UK, 1999.

[4] Mao W., Kincaid R. y Rifkin, A. – *On-line Algorithms for a Single Machine Scheduling Problem*, en Impact of Emerging Technologies in Computer Science and Operation Research, págs. 157 – 173, 1995.

[5] Lasso, M., Pandolfi D., De San Pedro M., Villagra A., Vilanova G., Gallard, R. – *Algorithms to Solve the Dynamic Weighted Tardiness Problems*, VIII Congreso Argentino de Ciencias de la Computación, Bs. As., pág. 609 – 616, Octubre 2002.

[6] Lasso, M., Pandolfi D., De San Pedro M., Villagra A., Gallard, R. – *Heuristics to Solve Dynamic W-T Problems in Single Machine Environments*, Proceedings of the International Conference on Computer Science, Software Engineering Information Technology, e-Business and Applications, págs. 432 –437, Rio de Janeiro, June 2003, Brazil.

[7] De San Pedro M., Lasso, M., Villagra A., Pandolfi D., Gallard, R. – *Solutions to the Dynamic Average Tardiness Problem in Single Machine Environmets*, IX Congreso Argentino de Ciencias de la Computación, La Plata, págs. 729 – 739, Octubre 2003.

[8] Lasso, M., Pandolfi D., De San Pedro M., Villagra A., , Gallard, R. – *Solving Dynamic Tardiness Problems in Single Machine Environments*, IEEE Congress on Evolutionary Computation, Vol. I, págs. 1143 – 1149, Portland, USA, 2004.

[9] Dorigo, M. y Stützle, T. – *Ant Colony Optimization*, The MIT Press, 2004.

[10] Branke J. – *Evolutionary Optimization in Dynamic Environments (Genetic Algorithms and Evolutionary Computation)*. Kluwer Academic Publishers (KAP), 2002.

[11] Colorni, A., Dorigo, M., Maniezzo V., Trubian, M. – *Ant System for Job Shop Scheduling*, JORBEL – Belgian Journal of Operations Research, Statistics and Computer Science, Vol. 34, N°19, págs. 39 – 53.

[12] Bauer, A., Bullheimer, B., Hartl R. F., Strauss, C. – *Minimizing Total Tardiness on a Single Machine using Ant Colony Optimization*, Central European Journal of Operations Research and Economics, Vol 8, N° 2, págs. 125-141, 2000.

[13] Blum C. – *ACO Applied to Group Shop Scheduling: A Case Study on Intensification and Diversification*, Proceedings of ANTS 2002, Lecture Notes in Computer Science Series, N° 2463, Springer-Verlag, 2002.

[14] Pfahringer, B. – *Multi-agent Search for Open Shop Scheduling. Adapting the Ant_Q Formalism*, Technical Report TR-96-09, Australian Research Institute for Artificial Intelligence, Vienna, 1996.

[15] Gagné C., Price, W. L., Gravel, M. – *Comparing an ACO Algorithm with other Metaheuristics for the Single Machine Scheduling Problem with Sequence-dependent setup times*, Journal of Operational Research Society, Vol. 53, págs. 895 – 906, 2002.

[16] Merkle, D., Middendorf, M. – *On Solving Permutation Scheduling Problems with Ant Colony Optimization*, Technical Report 415, Institute AIFB, University of Karlsruhe, 2002.

[17] Guntsch, M., Middendorf, M. – *Applying Population Based ACO to Dynamic Optimization Problems*, Proceedings of ANTS2002, Lecture Notes in Computer Science Series N° 2463, págs. 111 – 122, Springer-Verlag, 2002.

[18] de San Pedro M., Pandolfi D., Lasso M., Villagra A. - *Dynamic Scheduling Approaches to solve Single Machine Problems* – ASC 2005 - The Ninth IASTED International Conference on Artificial Intelligence and Soft Computing – pp 275-280, Benidorm, Spain, 2005

[19] de San Pedro M., Pandolfi D., Villagra A., Lasso M. – *Información heurística en ACO aplicada al Problema de Máquina Única de Tardanza Ponderada* – XII RPIC – Reunión de Trabajo en procesamiento de la Información y Control – pp 22, Río Gallegos, Santa Cruz, Argentina, 2007